A Logic Programming Approach to Behaviour Recognition

Alexander Artikis and Georgios Paliouras
Institute of Informatics & Telecommunications,
National Centre for Scientific Research "Demokritos",
Athens 15310, Greece
{a.artikis, paliourg}@iit.demokritos.gr

Abstract

We present a system for recognising human behaviour given a symbolic representation of surveillance videos. The input of our system is a set of time-stamped short-term behaviours, that is, behaviours taking place in a short period of time — walking, running, standing still, etc — detected on video frames. The output of our system is a set of recognised long-term behaviours — fighting, leaving an object, collapsing, etc — which are pre-defined temporal combinations of short-term behaviours. We develop a logic programming implementation of the Event Calculus to express the constraints on the short-term behaviours that, if satisfied, lead to the recognition of a long-term behaviour. We present experimental results concerning videos with several humans and objects, temporally overlapping and repetitive behaviours. Moreover, we show that our approach, which is not restricted to video surveillance, is better suited to a class of recognition applications than state-of-the-art recognition systems.

1 Introduction

We address the problem of human behaviour recognition by separating low-level recognition from high-level recognition. The output of the former type of recognition is a set of activities taking place in a short period of time — 'short-term behaviours'. The output of the latter type of recognition is a set of 'long-term behaviours', that is, pre-defined temporal combinations of short-term behaviours. We focus on high-level recognition.

To perform high-level recognition we define a set of long-term behaviours of interest — for example, 'leaving an object', 'fighting' and 'meeting' — as temporal combinations of short-term behaviours — for instance, 'walking', 'running', 'inactive' (standing still) and 'active' (body movement in the same position). We employ a logic programming implementation of the Event Calculus (EC) [9], a temporal reasoning formalism, in order to express the definition of a long-term behaviour. More precisely, we employ EC to express the temporal constraints on a set of short-term behaviours that, if satisfied, lead to the recognition of a long-term behaviour.

We evaluate our approach using an existing dataset of short-term behaviours detected on a series of surveillance videos. We recognise five long-term behaviours achieving high Recall and Precision rates.

Our behaviour recognition system falls under the category of *symbolic scenario recognition systems* (SSRS) — see, for example, [7, 6, 11, 18, 14, 13]. These systems accept as input a stream of time-stamped 'low-level' (or 'primitive') events, which are used to recognise 'high-level' (or 'derived') events of interest. The definition of a high-level event, included in a SSRS, imposes a set of constraints on a set of subevents. We present a comparison of our behaviour recognition system with state-of-the-art SSRS. We show that a logic programming approach is better suited to a class of recognition applications, including video surveillance, rather than purely temporal reasoning SSRS. Moreover, we show the advantages of our approach with respect to a state-of-the-art SSRS based on a logic programming implementation of an EC dialect.

The remainder of the paper is organised as follows. First, we present the Event Calculus. Second, we describe the dataset of short-term behaviours on which we perform long-term behaviour recognition. Third, we present our knowledge base of long-term behaviour definitions. Fourth, we present our experimental results. Finally, we discuss related work and outline directions for further research.

2 The Event Calculus

Our system for long-term behaviour recognition (LTBR) is a logic programming implementation of an Event Calculus formalisation, expressing long-term behaviour definitions. The Event Calculus (EC), introduced by Kowalski and Sergot [9], is a formalism for representing and reasoning about actions or events and their effects. We present

Table 1. Main Predicates of the Event Calculus.

Predicate	Meaning
happens(Act, T)	Action Act occurs at time T
initially$(F = V)$	The value of fluent F is V at time 0
holdsAt$(F = V, T)$	The value of fluent F is V at time T
holdsFor$(F = V, I)$	The value of fluent F is V during intervals I
initiates$(Act, F = V, T)$	The occurrence of action Act at time T initiates a period of time for which the value of fluent F is V
terminates$(Act, F = V, T)$	The occurrence of action Act at time T terminates a period of time for which the value of fluent F is V

here the version of the EC that we employ (more details about this version may be found in [3]).

EC is based on a many-sorted first-order predicate calculus. For the version used here, the underlying time model is linear and it may include real numbers or integers. Where F is a *fluent* — a property that is allowed to have different values at different points in time — the term $F = V$ denotes that fluent F has value V. Boolean fluents are a special case in which the possible values are true and false. Informally, $F = V$ holds at a particular time-point if $F = V$ has been *initiated* by an action at some earlier time-point, and not *terminated* by another action in the meantime.

An *action description* in EC includes axioms that define, among other things, the action occurrences (with the use of the happens predicate), the effects of actions (with the use of the initiates and terminates predicates), and the values of the fluents (with the use of the initially, holdsAt and holdsFor predicates). Table 1 summarises the main EC predicates. Variables (starting with an upper-case letter) are assumed to be universally quantified unless otherwise indicated. Predicates, function symbols and constants start with a lower-case letter.

The domain-independent definition of the holdsAt predicate is as follows:

$$\begin{aligned} &\text{holdsAt}(\ F = V,\ T\ ) \leftarrow \\ &\quad \text{initially}(\ F = V\ ), \\ &\quad \text{not broken}(\ F = V,\ 0,\ T\ ) \end{aligned} \tag{1}$$

$$\begin{aligned} &\text{holdsAt}(\ F = V,\ T\ ) \leftarrow \\ &\quad \text{happens}(\ Act,\ T'\ ),\ T' < T, \\ &\quad \text{initiates}(\ Act,\ F = V,\ T'\ ), \\ &\quad \text{not broken}(\ F = V,\ T',\ T\ ) \end{aligned} \tag{2}$$

According to axiom (1) a fluent holds at time T if it held initially (time 0) and has not been 'broken' in the meantime, that is, terminated between times 0 and T. Axiom (2) specifies that a fluent holds at a time T if it was initiated at some earlier time T' and has not been terminated between T' and T. 'not' represents 'negation by failure' [4]. The auxiliary predicate broken is defined as follows:

$$\begin{aligned} &\text{broken}(\ F = V,\ T_1,\ T_3\ ) \leftarrow \\ &\quad \text{happens}(\ Act,\ T_2\ ), \\ &\quad T_1 \leq T_2,\ T_2 < T_3, \\ &\quad \text{terminates}(\ Act,\ F = V,\ T_2\ ) \end{aligned} \tag{3}$$

$F = V$ is 'broken' between T_1 and T_3 if an event takes place in that interval that terminates $F = V$. Note that, according to the above axioms, a fluent does not hold at the time that was initiated but holds at the time it was terminated.

A fluent cannot have more than one value at any time. The following axiom captures this feature:

$$\begin{aligned} &\text{terminates}(\ Act,\ F = V,\ T\ ) \leftarrow \\ &\quad \text{initiates}(\ Act,\ F = V',\ T\ ), \\ &\quad V \neq V' \end{aligned} \tag{4}$$

Axiom (4) states that if an action Act initiates $F = V'$ then Act also terminates $F = V$, for all other possible values V of the fluent F. We do not insist that a fluent must have a value at every time-point. In this version of EC, therefore, there is a difference between initiating a Boolean fluent $F =$ false and terminating $F =$ true: the first implies, but is not implied by, the second.

The intervals in which a fluent holds are computed with the use of the holdsFor predicate. Below is a skeleton of this predicate:

$$\begin{aligned} &\text{holdsFor}(\ F = V,\ I\ ) \leftarrow \\ &\quad \text{start}(\ F = V,\ StartPts\ ), \\ &\quad \text{end}(\ F = V,\ EndPts\ ), \\ &\quad \text{compute\_intervals}(\ StartPts,\ EndPts,\ I\ ) \end{aligned} \tag{5}$$

The start predicate computes a list of time-points in which $F = V$ is initiated. If $F = V$ held initially then the output of start includes 0. The end predicate computes a list of time-points in which $F = V$ is terminated. Given the output of these predicates, compute_intervals computes the maximal intervals of time-points for which $F = V$ holds continuously. The computed intervals are of the form $(T_1, T_2]$ or since(T). To save space we do not present here the complete formalisation of holdsFor.

3 A Dataset of Short-Term Behaviours

LTBR includes an EC action description expressing long-term behaviour definitions. The input to LTBR is a

symbolic representation of short-term behaviours. The output of LTBR is a set of recognised long-term behaviours. In [2] we used the first dataset of the CAVIAR project[1] to perform long-term behaviour recognition. This dataset includes 28 surveillance videos of a public space. The videos are staged — actors walk around, browse information displays, sit down, meet one another, leave objects behind, fight, and so on. Each video has been manually annotated in order to provide the ground truth for both short-term and long-term behaviours. The CAVIAR dataset includes the following short-term behaviours: walking, running, active and inactive. We recognised the following long-term behaviours: a person leaving an object, a person being immobile, people meeting, moving together, or fighting.

Due to the absence of a short-term behaviour for abrupt motion, the accuracy of the recognition of the long-term behaviours meeting, moving and fighting was compromised. It was often impossible to distinguish between these behaviours — for instance, the short-term behaviours of people fighting were often classified as walking or active (due to the absence of an abrupt motion short-term behaviour), leading, in certain conditions, to the recognition of fighting and meeting, or fighting and moving. To overcome this problem, we introduced in the CAVIAR dataset a short-term behaviour for abrupt motion: we manually edited the annotation of the CAVIAR videos by changing, when necessary, the label of a short-term behaviour to 'abrupt motion'. A person is said to exhibit an abrupt motion behaviour if she moves abruptly and her position in the global coordinate system does not change significantly — if it did then her short-term behaviour would be classified as running. A definition of abrupt motion and a description of a system detecting this type of short-term behaviour may be found in [8], for example.

For this set of experiments, therefore, the input to LTBR is: (i) the short-term behaviours walking, running, active, inactive and abrupt motion, along with their time-stamps, that is, the frame in which a short-term behaviour took place, (ii) the coordinates of the tracked people and objects as pixel positions at each time-point, and (iii) the first time and the last time a person or object is tracked ('appears'/'disappears'). Given this input, LTBR recognises the following long-term behaviours: a person leaving an object, a person being immobile, people meeting, moving together, or fighting.

Short-term behaviours are represented as EC actions whereas the long-term behaviours that LTBR recognises are represented as EC fluents. In the following section we present example fragments of the long-term behaviour definitions. The complete definitions are available with the source code of LTBR.

[1] http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

4 Long-Term Behaviour Definitions

The long-term behaviour 'leaving an object' is defined as follows:

$$\begin{aligned}&\mathsf{initiates}(\ inactive(Object),\\ &\quad leaving\_object(Person,\ Object) = \mathsf{true},\ T\ ) \leftarrow\\ &\qquad \mathsf{holdsAt}(\ appearance(Object) = appear,\ T\ ),\\ &\qquad \mathsf{holdsAt}(\ close(Person,\ Object,\ 30) = \mathsf{true},\ T\ ),\\ &\qquad \mathsf{holdsAt}(\ appearance(Person) = appear,\ T_0\ ),\\ &\qquad T_0 < T\end{aligned} \tag{6}$$

$$\begin{aligned}&\mathsf{initiates}(\ exit(Object),\\ &\quad leaving\_object(Person,\ Object) = \mathsf{false},\ T\ )\end{aligned} \tag{7}$$

Axiom (6) expresses the conditions in which a 'leaving an object' behaviour is recognised. The fluent recording this behaviour, $leaving\_object(Person,\ Object)$, becomes true at time T if $Object$ is inactive at T, $Object$ 'appears' at T, there is a $Person$ close to $Object$ at T (in a sense to be specified below), and $Person$ has appeared at some time earlier than T. The $appearance$ fluent records the times in which an object/person 'appears' and 'disappears'. The $close(A, B, D)$ fluent is true when the distance between A and B is at most D. The distance between two tracked objects/people is computed given their coordinates. Based on our empirical analysis the distance between a person leaving an object and the object is at most 30 pixel positions.

An object exhibits only inactive short-term behaviour. Any other type of short-term behaviour would imply that what is tracked is not an object. Therefore, the short-term behaviours active, walking, running and abrupt motion do not initiate the $leaving\_object$ fluent. In the CAVIAR videos an object carried by a person is not tracked — only the person that carries it is tracked. The object will be tracked, that is, 'appear', if and only if the person leaves it somewhere. Consequently, given axiom (6), the $leaving\_object$ behaviour will be recognised only when a person leaves an object (see the third line of axiom (6)), not when a person carries an object.

Axiom (7) expresses the conditions in which a $leaving\_object$ behaviour ceases to be recognised. In brief, $leaving\_object$ is terminated when the object in question is picked up. $exit(A)$ is an event that takes place when $appearance(A) = disappear$. An object that is picked up by someone is no longer tracked — it 'disappears' — triggering an $exit$ event which in turn terminates $leaving\_object$.

The long-term behaviour $immobile$ was defined in order to signify that a person is resting in a chair or on the floor, or has fallen on the floor (fainted, for example). Below is

an axiom of the $immobile$ definition:

$$\begin{array}{l}\mathsf{initiates}(\ inactive(Person),\\ \quad immobile(Person) = \mathsf{true},\ T\ ) \leftarrow\\ \qquad \mathsf{duration}(\ inactive(Person),\ Intervals\ ),\\ \qquad (T,\ T_1)\ \in\ Intervals,\\ \qquad T_1 > T{+}54,\\ \qquad \mathsf{findall}(\ S,\ shop(S),\ Shops\ ),\\ \qquad far(\ Person,\ Shops\ ),\\ \qquad \mathsf{happens}(\ active(Person),\ T_0\ ),\\ \qquad T_0 < T\end{array} \tag{8}$$

According to axiom (8), the behaviour $immobile(Person)$ is recognised if:

- $Person$ stays inactive for more than 54 frames (see lines 3–5 of axiom (8)). We chose this number of frames given our empirical analysis of the CAVIAR dataset. duration is a predicate computing the duration of inactive behaviour given the number of consecutive instantaneous inactive events. The output of duration is a set of the maximal intervals in which a person/object is inactive. (holdsFor computes the duration of fluents and thus cannot be used for computing the duration of inactive behaviour.) Note that this is not the only way to represent durative events in EC. See [16] for alternative representations.

- $Person$ is not close to an information display or a shop (see lines 6–7 of axiom (8)). If $Person$ was close to a shop then she would have to stay inactive much longer than 54 frames before $immobile$ could be recognised. In this way we avoid classifying the behaviour of browsing a shop as $immobile$. $far(P, List)$ is an atemporal predicate that becomes true when P is far from every element of the $List$.

- $Person$ has been active some time in the past (see lines 8–9 of axiom (8)). The definition of $immobile$ includes axioms requiring that $Person$ has been walking some time in the past (in addition to the above constraints). We insist that $Person$ in $immobile(Person)$ has been active or walking before being inactive in order to distinguish between a left object, which is inactive from the first time it is tracked, from an immobile person.

$immobile(Person)$ is terminated when $Person$ starts walking, running or 'disappears' — see axioms (9)–(11):

$$\begin{array}{l}\mathsf{initiates}(\ walking(Person),\\ \quad immobile(Person) = \mathsf{false},\ T\ )\end{array} \tag{9}$$

$$\begin{array}{l}\mathsf{initiates}(\ running(Person),\\ \quad immobile(Person) = \mathsf{false},\ T\ )\end{array} \tag{10}$$

$$\begin{array}{l}\mathsf{initiates}(\ exit(Person),\\ \quad immobile(Person) = \mathsf{false},\ T\ )\end{array} \tag{11}$$

In a similar way we may express the definitions of other long-term behaviours. It is not difficult to see that the use of EC, in combination with the full power of logic programming, allows us to express behaviour definitions including complex temporal, spatial or other constraints. Below we present fragments of the remaining behavior definitions of LTBR's knowledge base.

The following axioms represent a fragment of the $moving$ behaviour definition:

$$\begin{array}{l}\mathsf{initiates}(\ walking(Person),\\ \quad moving(Person,\ Person_2) = \mathsf{true},\ T\ ) \leftarrow\\ \qquad \mathsf{holdsAt}(\ close(Person,\ Person_2, 34) = \mathsf{true},\ T\ ),\\ \qquad \mathsf{happens}(\ walking(Person_2),\ T\ )\end{array} \tag{12}$$

$$\begin{array}{l}\mathsf{initiates}(\ walking(Person),\\ \quad moving(Person,\ Person_2) = \mathsf{false},\ T\ ) \leftarrow\\ \qquad \mathsf{holdsAt}(\ close(Person,\ Person_2,\ 34) = \mathsf{false},\ T\ )\end{array} \tag{13}$$

$$\begin{array}{l}\mathsf{initiates}(\ active(Person),\\ \quad moving(Person,\ Person_2) = \mathsf{false},\ T\ ) \leftarrow\\ \qquad \mathsf{happens}(\ active(\ Person_2\ ),\ T\ )\end{array} \tag{14}$$

$$\begin{array}{l}\mathsf{initiates}(\ running(Person),\\ \quad moving(Person,\ Person_2) = \mathsf{false},\ T\ )\end{array} \tag{15}$$

$$\begin{array}{l}\mathsf{initiates}(\ abrupt(Person),\\ \quad moving(Person,\ Person_2) = \mathsf{false},\ T\ )\end{array} \tag{16}$$

$$\begin{array}{l}\mathsf{initiates}(\ exit(Person),\\ \quad moving(Person,\ Person_2) = \mathsf{false},\ T\ )\end{array} \tag{17}$$

According to axiom (12), $moving$ is initiated when two people are walking and are close to each other (their distance is at most 34 pixel positions). $moving$ is terminated when the people walk away from each other, that is, their distance becomes greater than 34 pixel positions (see axiom (13)), when they stop moving, that is, become active (see axiom (14)) or inactive, when one of them starts running (see axiom (15)), moving abruptly (see axiom (16)), or 'disappears' (see axiom (17)).

$meeting$ is recognised when the following conditions are satisfied:

$$\begin{array}{l}\mathsf{initiates}(\ active(Person),\\ \quad meeting(Person,\ Person_2) = \mathsf{true},\ T\ ) \leftarrow\\ \qquad \mathsf{holdsAt}(\ close(Person,\ Person_2,\ 25) = \mathsf{true},\ T\ ),\\ \qquad \mathsf{not\ happens}(\ running(Person_2),\ T\ ),\\ \qquad \mathsf{not\ happens}(\ abrupt(Person_2),\ T\ )\end{array} \tag{18}$$

$$\begin{aligned}&\text{initiates}(\ inactive(Person),\\ &\quad meeting(Person,\ Person_2) = \text{true},\ T\ ) \leftarrow\\ &\qquad \text{holdsAt}(\ close(Person,\ Person_2,\ 25) = \text{true},\ T\ ),\\ &\qquad \text{not happens}(\ running(Person_2),\ T\ ),\\ &\qquad \text{not happens}(\ abrupt(Person_2),\ T\ )\end{aligned} \tag{19}$$

$meeting$ is initiated when two people 'interact': at least one of them is active or inactive, the other is neither running nor moves abruptly, and the distance between them is at most 25 pixel positions. This interaction phase can be seen as some form of greeting (for example, a handshake). $meeting$ is terminated when the two people walk away from each other, or one of them starts running, moving abruptly or 'disappears'. The axioms representing the termination of $meeting$ are similar to axioms (13) and (15)–(17).

Note that $meeting$ may overlap with $moving$: two people interact and then start $moving$, that is, walk while being close to each other. In general, however, there is no fixed relationship between $meeting$ and $moving$.

The last definition of LTBR's knowledge base concerns the *fighting* behaviour — the axiom below presents the conditions in which *fighting* is initiated:

$$\begin{aligned}&\text{initiates}(\ abrupt(Person),\\ &\quad fighting(Person,\ Person_2) = \text{true},\ T\ ) \leftarrow\\ &\qquad \text{holdsAt}(\ close(Person,\ Person_2,\ 24) = \text{true},\ T\ ),\\ &\qquad \text{not happens}(\ inactive(Person_2),\ T\ )\end{aligned} \tag{20}$$

Two people are assumed to be *fighting* if at least one of them is moving abruptly, the other is not inactive, and the distance between them is at most 24 pixel positions. *fighting* is terminated when one of the people walks or runs away from the other, or 'disappears'.

5 Experimental Results

We present our experimental results on 28 surveillance videos of the CAVIAR project. These videos contain 26419 frames that have been manually annotated in order to provide the ground truth for short-term and long-term behaviours. We edited the original annotation of the CAVIAR videos by introducing a short-term behaviour for abrupt motion. (The edited annotation of the CAVIAR videos is available with the source code of LTBR.) Table 2 shows the performance of LTBR — we show, for each long-term behaviour, the number of True Positives (TP), False Positives (FP) and False Negatives (FN), as well as Recall and Precision.

LTBR correctly recognised 4 *leaving_object* behaviours. Moreover, there were no FP. On the other hand, there was 1 FN. This, however, cannot be attributed to LTBR because in the video in question the object was left behind a chair and was not tracked. In other words, the left object never 'appeared', it never exhibited a short-term behaviour.

Table 2. Experimental Results.

Behaviour	TP	FP	FN	Recall	Precision
leaving object	4	0	1	0.8	1
immobile	9	8	0	1	0.52
moving	15	3	2	0.88	0.83
meeting	6	1	3	0.66	0.85
fighting	6	0	0	1	1

Regarding $immobile$ we had 9 TP, 8 FP and no FN. The recognition of $immobile$ would be much more accurate if there was a short-term behaviour for the motion of leaning towards the floor or a chair. Due to the absence of such a short-term behaviour, the recognition of $immobile$ is primarily based on how long a person is inactive. In the CAVIAR videos a person falling on the floor or resting in a chair stays inactive for at least 54 frames. Consequently LTBR recognises $immobile$ if, among other things, a person stays inactive for at least 54 frames (we require that a person stays inactive for a longer time period if she is located close to a shop to avoid FP when a person is staying inactive browsing a shop). There are situations, however, in which a person stays inactive for more than 54 frames and has not fallen on the floor or sat in a chair: people watching a fight, or just staying inactive waiting for someone. It is in those situations that we have the FP concerning $immobile$. We expect that in longer videos recording actual behaviours (as opposed to the staged behaviours of the CAVIAR videos) a person falling on the floor or resting in a chair would be inactive longer than a person staying inactive while standing. In this case we could increase the threshold for the duration of inactive behaviour in the definition of $immobile$, thus potentially reducing the number of FP concerning $immobile$.

The introduction of the abrupt motion short-term behaviour had no effect on LTBR's accuracy of the $leaving\_object$ and $immobile$ behaviour recognition.

LTBR recognised correctly 15 $moving$ behaviours. However, it also recognised incorrectly 3 such behaviours. These FP mainly concern people that do move together: walk towards the same direction while being close to each other. According to the manual annotation of the videos, however, these people do not exhibit the $moving$ long-term behaviour. (See [10] for an evaluation of the manual annotation of the CAVIAR dataset.)

The number of FP concerning the $moving$ behaviour has substantially decreased with respect to the results presented in [2]. One reason for reducing these FP concerns the fact that we added a constraint to the EC action description of LTBR that the duration of $moving$ exceeds a specified

threshold. Consequently LTBR did not classify as *moving* the cases in which people happen to walk close to each other while moving to different directions.

Another reason for reducing the FP regarding *moving* is the introduction of abrupt motion. In the original annotation of the CAVIAR dataset the short-term behaviours of people *fighting* were sometimes classified as walking. Consequently, the behaviour of these people was incorrectly recognised by LTBR as *moving*, since, according to the original annotation of the CAVIAR dataset, they are walking while being close to each other (moreover, their coordinates changed). Labelling the short-term behaviours of people *fighting* as abrupt motion resolves this issue, because abrupt motion does not initiate a *moving* behaviour.

LTBR did not recognise 2 *moving* behaviours. One FN was due to the fact that the distance between the people walking together was greater than the threshold we have specified. Increasing this threshold would result in substantially increasing the number of FP. Therefore we chose not to increase it. The other FN was due to the constraint that we added in the EC action description of LTBR that the duration of *moving* should exceed a specified threshold — a *moving* behaviour took place having duration less than this threshold. We chose to keep this contraint nevertheless because it substantially reduces the number of FP.

LTBR recognised 7 *meeting* behaviours, 6 of which took place and 1 did not take place. The FP was due to the fact that two people were active and close to each other, but were not interacting. LTBR did not recognise 3 *meeting* behaviours. 2 FN were due to the fact that the distance between the people in the meeting was greater than the threshold we have specified. If we increased that threshold LTBR would correctly recognise these 2 *meeting* behaviours. However, the number of FP for *meeting* would substantially increase. Therefore we chose not to increase the threshold distance. The third FN was due to the fact that the short-term behaviours of the people interacting — handshaking — were classified as walking (although one of them was actually active). We chose to specify that walking does not initiate a *meeting* in order to avoid incorrectly recognising meetings when people simply walk close to each other.

In the original annotation of the CAVIAR dataset the short-term behaviours of people *fighting* were sometimes classified as active. Consequently, in these cases LTBR incorrectly recognised the *meeting* behaviour. The introduction of the abrupt motion short-term behaviour in the CAVIAR dataset reduced the number of FP concerning *meeting* — the short-term behaviours of people *fighting* are now classified as abrupt motion thus not initiating a *meeting*.

LTBR's recognition accuracy with respect to *fighting* was perfect, that is, there were no FP or FN. The previous version of LTBR had 8 FP and 2 FN regarding *fighting*. The increased accuracy is due to the introduction of abrupt motion in the CAVIAR dataset and the corresponding modification of the *fighting* definition in LTBR — now only abrupt motion initiates *fighting* whereas before active and running short-term behaviours initiated *fighting*. In this version of LTBR there is no confusion between *fighting* and *meeting* — consequently, there are no FP now for *fighting* when a *meeting* takes place. Moreover, we now avoid the FN concerning *fighting* that were due to the fact that the short-term behaviours of people *fighting* were (sometimes) classified as walking — these behaviours are now classified as abrupt motion.

Overall the introduction of abrupt motion in the dataset and the corresponding modification of the long-term behaviour definitions substantially increased the recognition accuracy of LTBR. Behaviours that could be, and were confused in the past, such as *fighting* and *meeting*, and *fighting* and *moving*, are no longer confused. The introduction of abrupt motion, however, could, in some cases, reduce the recognition accuracy of LTBR. This would happen when a person moved abruptly (say fainted and fell on the floor) while being close to another person that was not inactive at the time. In this case LTBR would incorrectly recognise *fighting*. Such a combination of short-term behaviours does not take place in the CAVIAR dataset.

6 Related Work

A well-known system for behaviour recognition is the Chronicle Recognition System (CRS)[2]. A 'chronicle' can be seen as a long-term behaviour. In order to compare LTBR with CRS, we expressed the long-term behaviour definitions presented in this paper in the CRS input language. Below, for instance, is a fragment of the `immobile` definition in the CRS language. An `event(e, T)` predicate expresses the occurrence of an event `e` at time-point `T`, `occurs(n,m,e,(T1, T2))` expresses that event `e` should take place at least `n` times and at most `m` times in the interval [`T1, T2`), `noevent(e,(T1, T2))` expresses that event `e` should not take place in the interval [`T1, T2`), while `hold(f:v,(T1, T2))` expresses that the value of fluent `f` should be `v` during the interval [`T1, T2`). `?` is the prefix of an atemporal variable. Details about the CRS language and CRS in general may be found on the web page of the system and in [7, 5, 6].

```
(1) chronicle immobileInitiate() {
(2)   occurs( 54, 54,
              st[inactive,?id,?x,?y],
              (T1, T1+54) )
(3)   noevent( st[inactive,?id,*,*],
               (T1-1, T1) )
```

[2] `http://crs.elibel.tm.fr/`

```
(4)   hold( shop[shop1,?sX,?sY]:true,
            (T1, T1+1) )
(5)   ?x != ?sX or ?y != ?sY
(6)   event( st[active,?id,*,*], T0 )
(7)   T0 < T1
(8)   noevent( st[active,?id,*,*],
               (T0+1, T1) )
(9)   when recognized {
(10)   emit event( initiateImmobile[?id],
                   T1 )
(11)  }
(12)}
```

`st[?sbeh, ?id, ?x, ?y]` denotes that `?id` has the short-term behaviour `?sbeh` and her coordinates are (`?x, ?y`). `shop[?s, ?x, ?y]` denotes the coordinates (`?x, ?y`) of shop `?s`. The above fragment of the `immobile` definition expresses the conditions in which `immobile` is initiated. In LTBR these conditions are expressed by axiom (8). In brief, an `immobile(?id)` behaviour is recognised if `?id` stays inactive for more than 54 frames (see lines 2–3), and has been active some time in the past (see lines 6–8). If the conditions of the above chronicle are satisfied, an event is triggered, denoting that `immobile` is initiated (see lines 9–11). The triggered event is used by another chronicle computing the duration of an `immobile` behaviour.

Recall that there was an additional constraint to the recognition of `immobile`: the person in question must be 'far' from all shops — if the person is close to a shop then she would have to stay inactive much longer than 54 frames before `immobile` could be recognised. This constraint cannot be expressed in the CRS language for two reasons. First, the CRS language does not allow for the computation of the distance between two people/objects. This is due to the fact that no mathematical operators are allowed in the constraints of atemporal variables. Our formalisation of `immobile` in the CRS language requires that the `x` coordinate or the `y` coordinate of the person in question is different from the respective coordinate (`sX` or `sY`) of a shop (see line 5). Clearly, this is an inappropriate formalisation of the constraint that a person is far from a shop. Second, it is not possible to express that a person is 'far' from *all* shops due to the limitations of the CRS language concerning universal quantification.

CRS, therefore, cannot be directly used for behaviour recognition in video surveillance applications. Moreover, CRS cannot be directly used for behaviour recognition in any application requiring any form of spatial reasoning, or any other type of atemporal reasoning. These limitations could be overcome by developing a separate tool for atemporal reasoning that would be used by CRS whenever this form of reasoning was required. To the best of our knowledge, such extensions of CRS are not available. (Clearly, the computational efficiency of CRS, which is one of the main attractions of using this system for behaviour recognition, would be compromised by the integration of an atemporal reasoner.)

In our approach to behaviour recognition, the availability of the full power of logic programming, which is one of the main attractions of employing the Event Calculus (EC) as the temporal formalism, allows for the development of behaviour definitions including complex temporal (EC is at least as expressive as the CRS language with respect to temporal representation) and atemporal constraints.

We do not present a comparison of the computational performance of LTBR and CRS — such a comparison is meaningless due to the fact the behaviour definitions of LTBR are more complex (they include representations of the atemporal aspects of the definitions) than those of CRS.

Our approach to behaviour recognition has a formal, declarative semantics. This is in contrast to other behaviour recognition systems proposed in the literature ([11, 18, 17]). The employed version of EC allows for the development of a recognition system capable of dealing with, among other things, durative (short-term and long-term) behaviours, temporally overlapping, repetitive, and 'forbidden' behaviours, that is, behaviours that should not take place within a specified time-period in order to recognise some other behaviour. When necessary, more expressive EC versions may be employed (see [16, 12] for presentations of the EC expressiveness).

Paschke and colleagues have proposed a logic programming implementation of an EC version for behaviour recognition — see, for example, [14, 15, 13]. Unlike our EC version, there is no support for multi-valued fluents — only Boolean fluents are considered. The use of multi-valued fluents makes the representation considerably more succinct. Moreover, the EC version of Paschke and colleagues does not include axioms for recognising a behaviour that has been initiated at some earlier time-point and has not (yet) terminated. For example, there is no built-in support for recognising an on-going *fighting* behaviour. Our treatment of behaviours as EC fluents in combination with the holdsFor predicate for computing the intervals in which a fluent holds, allows us to overcome the above limitation. In the case of an on-going *fighting* behaviour, for instance, an answer to a query regarding *fighting* would be of the form since(T), indicating that the recognition of *fighting* started at T and has not (yet) ended.

7 Summary and Further Work

We presented a system for long-term behaviour recognition using a symbolic representation of short-term behaviours detected on surveillance videos. Our intuition that an explicit representation of a short-term behaviour for

abrupt motion, as opposed to implicitly representing abrupt motion by means of the active or walking behaviours, would significantly improve the recognition of a class of long-term behaviours, was experimentally verified.

Our behaviour recognition system, LTBR, falls under the category of symbolic scenario recognition systems. LTBR is a logic programming implementation of the Event Calculus (EC). We showed that a logic programming approach is better suited to a class of recognition applications, including video surveillance, rather than purely temporal reasoning systems. Moreover, we showed the advantages of our approach with respect to a state-of-the-art recognition system based on a logic programming implementation of an EC dialect.

A logic programming approach to behaviour recognition has the advantage that machine learning techniques can be directly employed in order to adapt a knowledge base of behaviour definitions with the aim of improving behaviour recognition accuracy. An area of current work is the use of inductive logic programming (ILP) techniques for fine-tuning in an automated way behaviour definitions (see, for example, [1] for an application of ILP techniques on EC formalisations).

Another direction of current research concerns the experimental validation of our approach to behaviour recognition in the fields of emergency rescue operations and public transport services. In the context of the EU-project PRONTO we will define and recognise long-term behaviours that take place in the aforementioned application domains with the aim of supporting intelligent resource management.

Acknowledgments

This work has been undertaken in the context of EU-funded PRONTO Project (FP7-ICT 231738). We would like to thank Christophe Dousson for his suggestions regarding the Chronicle Recognition System. The authors themselves, however, are solely responsible for any misunderstanding about the use of this technology. We should also like to thank Anastasios Skarlatidis for converting the XML representation of the CAVIAR dataset into an Event Calculus representation.

References

[1] D. Alrajeh, O. Ray, A. Russo, and S. Uchitel. Extracting requirements from scenarios with ILP. In *Inductive Logic Programming*, volume LNAI 4455. Springer, 2007.

[2] A. Artikis and G. Paliouras. Behaviour recognition using the event calculus. In *Proceednigs of IFIP Conference on Artificial Intelligence Applications & Innovations (AIAI)*. Springer, 2009.

[3] A. Artikis, M. Sergot, and J. Pitt. Specifying norm-governed computational societies. *ACM Transactions on Computational Logic*, 10(1), 2009.

[4] K. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Databases*, pages 293–322. Plenum Press, 1978.

[5] C. Dousson. Extending and unifying chronicle representation with event counters. In *Proceedings of European Conference on Artificial Intelligence (ECAI)*, pages 257–261. IOS Press, 2002.

[6] C. Dousson and P. L. Maigat. Chronicle recognition improvement using temporal focusing and hierarchisation. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 324–329, 2007.

[7] M. Ghallab. On chronicles: Representation, on-line recognition and learning. In *Proceedings of Conference on Principles of Knowledge Representation and Reasoning*, pages 597–606, 1996.

[8] D. Kosmopoulos, P. Antonakaki, k. Valasoulis, A. Kesidis, and S. Perantonis. Human behaviour classification using multiple views. In *Proceedings of Hellenic Conference on Artificial Intelligence*, 2008.

[9] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–96, 1986.

[10] T. List, J. Bins, J. Vazquez, and R. B. Fisher. Performance evaluating the evaluator. In *Proceedings of International Conference on Computer Communications and Networks*, pages 129–136. IEEE Computer Society, 2005.

[11] D. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley, 2002.

[12] R. Miller and M. Shanahan. The event calculus in a classical logic — alternative axiomatizations. *Journal of Electronic Transactions on Artificial Intelligence*, 4(16), 2000.

[13] A. Paschke. ECA-RuleML: An approach combining ECA rules with temporal interval-based KR event/action logics and transactional update logics. Technical Report 11, Technische Universität München, 2005.

[14] A. Paschke and M. Bichler. Knowledge representation concepts for automated SLA management. *Descision Support Systems*, 46(1):187–205, 2008.

[15] A. Paschke, A. Kozlenkov, and H. Boley. A homogeneous reaction rule language for complex event processing. In *Proceedings of International Workshop on Event-driven Architecture, Processing and Systems*, 2007.

[16] M. Shanahan. The event calculus explained. In M. Wooldridge and M. Veloso, editors, *Artificial Intelligence Today*, LNAI 1600, pages 409–430. Springer, 1999.

[17] V.-T. Vu. *Temporal Scenarios for Automatic Video Interpretation*. PhD thesis, Université de Nice — Sophia Antipolis, 2004.

[18] V.-T. Vu, F. Brémond, and M. Thonnat. Automatic video interpretation: A novel algorithm for temporal scenario recognition. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 1295–1302, 2003.